HW Class 06

Andres Rivero

10/16/2021

Can you improve this analysis code?

library(bio3d) s1 <- read.pdb("4AKE") # kinase with drug
s2 <- read.pdb("1AKE") # kinase no drug
s3 <- read.pdb("1E4Y") # kinase with drug

s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s1, chain="A", elety="CA")

s1.b <- s1.chainA$atomb
s2.b <- s2.chainA$atomb
s3.b <- s3.chainA$atomb

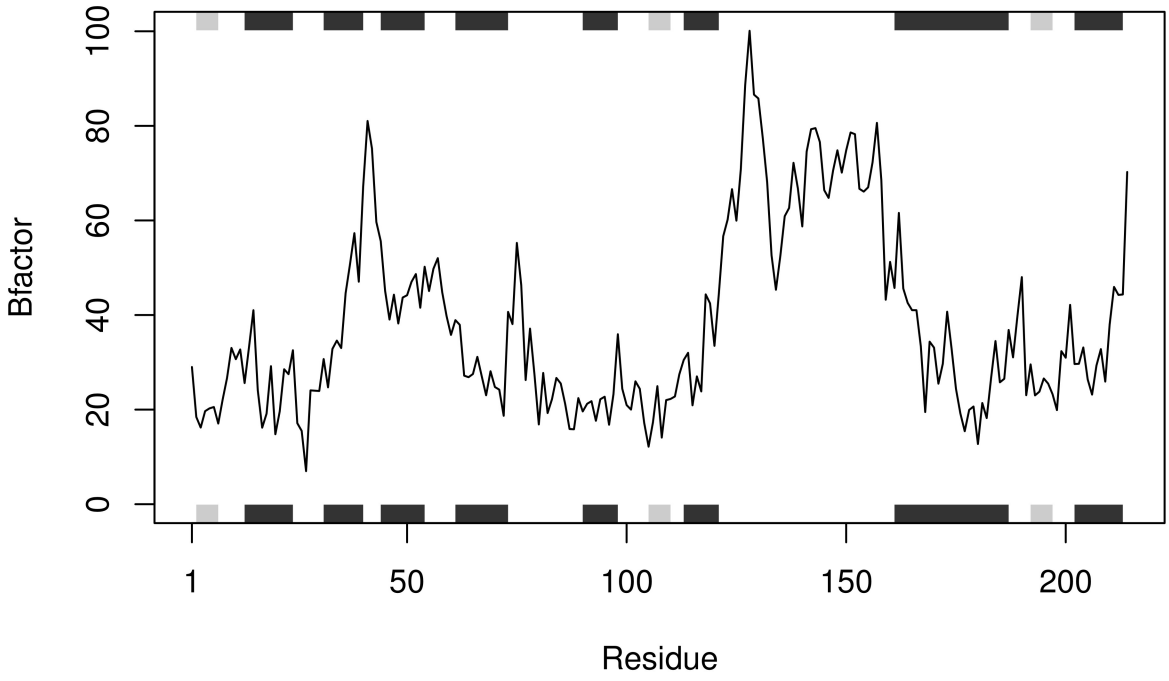
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
plotb3(s2.b, sse=s2.chainA, typ="l", ylab="Bfactor")
plotb3(s3.b, sse=s3.chainA, typ="l", ylab="Bfactor")

We will start by getting a snippet of each of the different codes ran for the different enzymes. What does this code do?

```
library(bio3d)
s1 <- read.pdb("4AKE") # kinase with drug
```

```
##   Note: Accessing on-line PDB file
```

```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s1.b <- s1.chainA$atom$b
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
```



It seem like the code looks the name of a given protein on a database, downloads information about it into the R environment, trims some of that info and gives a plot for said protein.

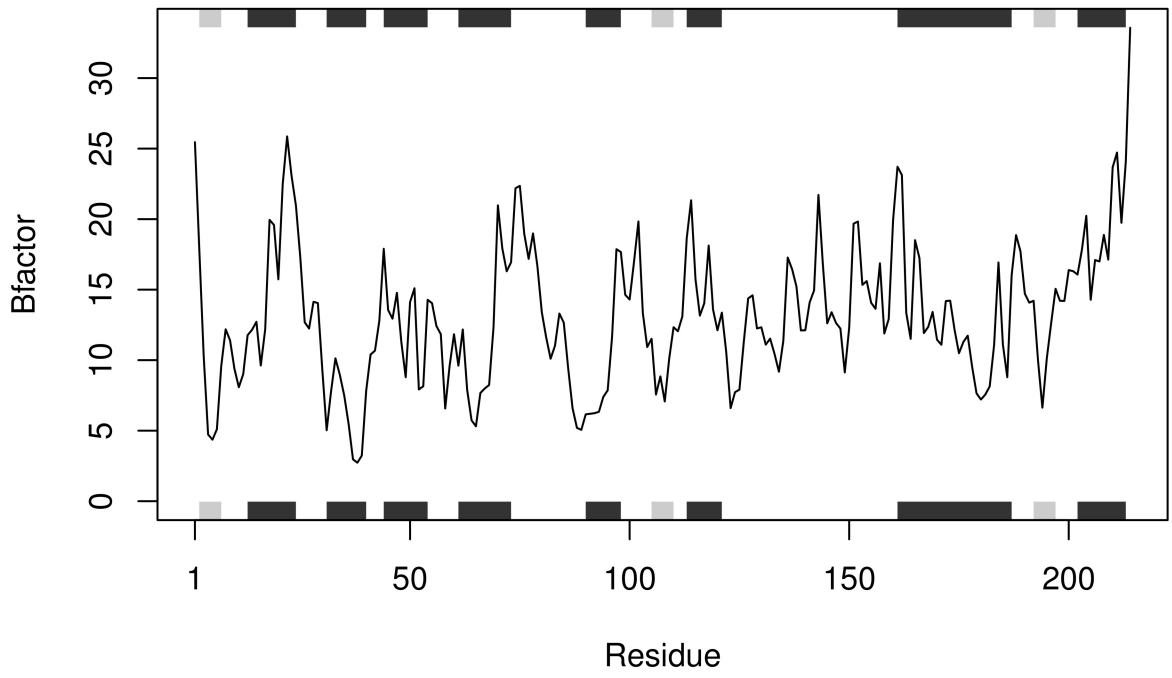
How could we simplify/or generalize the code shown above to make our function?
We could start by substituting "x" in place of

sx <- read.pdb(x) #This would get the info about "x" protein from the database
sx.chainA <- trim.pdb(sx, chain="A", elety="CA") #This would trim that information
sx.b <- sx.chainA*atom*b #This would select the part of all that information that we want to plot
plotb3(sx.b, sse=s1.chainA, typ="l", ylab="Bfactor") #And this would plot it

We could include that in the body of our function and see if it manages to plot a protein (we will try with the third one)

```
plotprot <- function(x){
  sx <- read.pdb(x)
sx.chainA <- trim.pdb(sx, chain="A", elety="CA")
sx.b <- sx.chainA$atom$b
plotb3(sx.b, sse=s1.chainA, typ="l", ylab="Bfactor")
}
plotprot("1E4Y")
```

```
##   Note: Accessing on-line PDB file
```

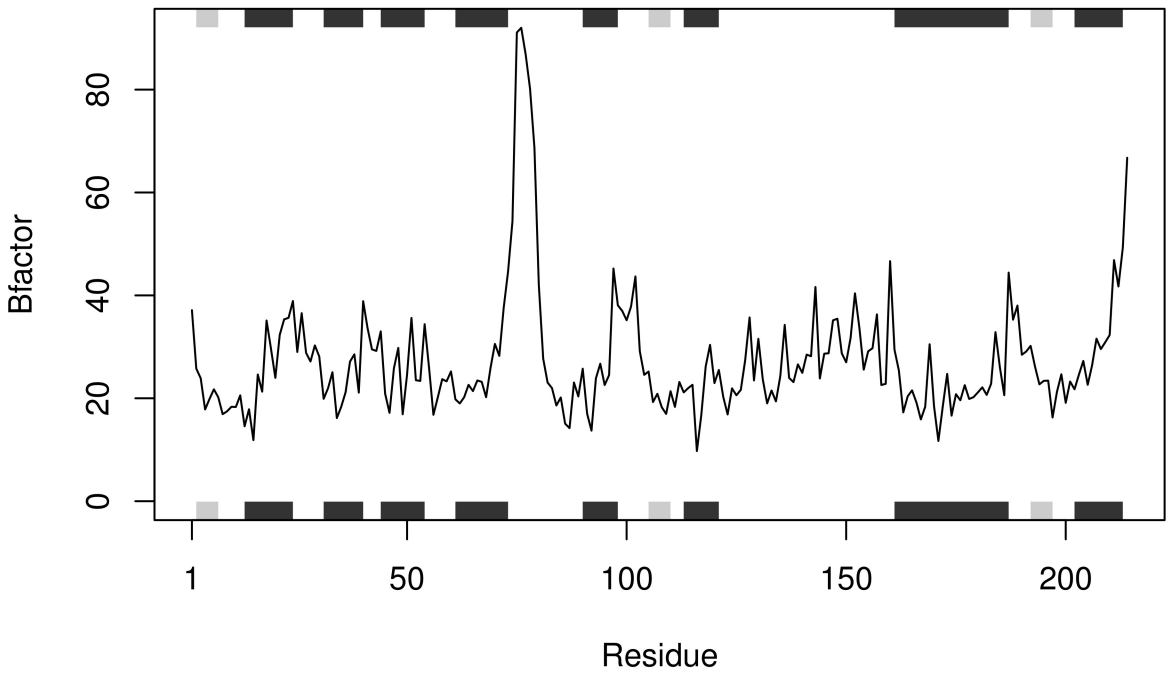


The function seemed to work.
By just giving it the protein code it automatically gives us the same plot that we were getting with 4 different lines of code before.

We will do another test with the second protein from the example:

```
plotprot("1AKE")
```

```
##   Note: Accessing on-line PDB file
##    PDB has ALT records, taking A only, rm.alt=TRUE
```

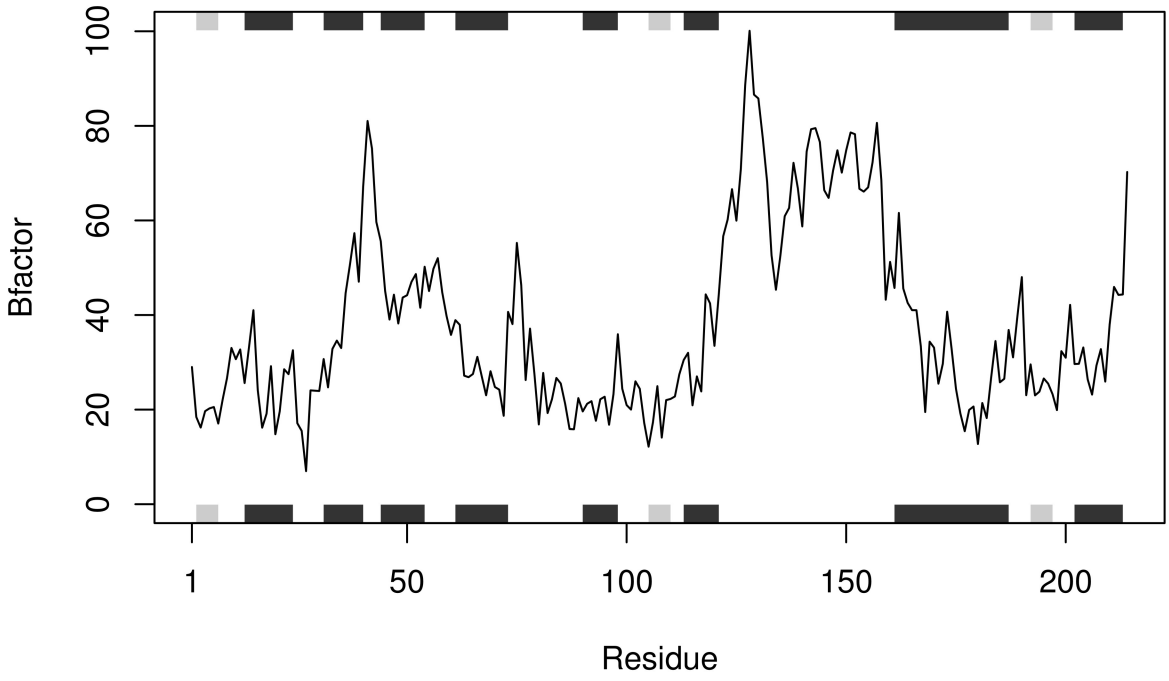


And a final test with the very first one, and compare it to the first graph obtained in this pdf:

```
plotprot("4AKE")
```

```
##   Note: Accessing on-line PDB file

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE): C:
## \Users\andre\AppData\Local\Temp\RtmpiKRLdD/4AKE.pdb exists. Skipping download
```



It worked and the plots look the same. **SUCCESS**